REMARKS

Claims 1 – 19 are pending and at issue in the present application.

The official action provisionally rejects claims 1 – 19 on the grounds of nonstatutory obviousness-type double patenting in light of claims 1 – 26 of copending Application Serial No. 10/395,557. The action rejects claims 1 – 19 under 35 U.S.C. § 112, ¶ 1, as failing to comply with the written description requirement. The action rejects claims 17 – 19 under 35 U.S.C. § 112, ¶ 2, as being indefinite. The action rejects claims 1 – 19 under 35 U.S.C. § 102 as anticipated by U.S. Patent No. 6,397,242 ("*Devine et al.*").

1. PROVISIONAL OBVIOUSNESS TYPE DOUBLE PATENTING REJECTION

Claims 1 – 19 stand provisionally rejected on the grounds of non-statutory obviousness-type double patenting over claims 1 – 26 of copending Application Serial No. 10/395,557. Because the rejection is provisional, applicant takes no action at this time in regards to this rejection.

2. REJECTION UNDER 35 U.S.C. § 112, ¶ 1

The office action rejects claims 1 – 19 under 35 U.S.C. § 112, ¶ 1 for purportedly failing to comply with the written description requirement. In particular, the office action points to paragraph [0038] of the application as making it unclear "which embodiments apply to the invention." The office action goes on to state that this paragraph could be construed to mean that the embodiments in the written description are not representative of the invention, but merely representative of parts of the invention.

This rejection has no basis in law or in fact and is respectfully, but strongly traversed.

First, the office action fails to provide the specificity required to meet the heightened showing the MPEP sets for issuing a written description rejection based on originally-filed claims. The MPEP instructs that "[t]here is a strong presumption that an adequate written description of the claimed invention is present when the application is filed." MPEP § 2163 II.A. (emphasis added). "Consequently, rejection of an original claim for lack of written description should be rare." MPEP § 2163 II.A. (emphasis added).

Second, in focusing on a single paragraph of the specification the examiner incorrectly ignores the entirety of the disclosure. Under 35 U.S.C. § 112, ¶ 1, possession, which notably may be shown by a <u>single</u> embodiment, is evidenced by reading the "entire application" (*see* MPEP § 2163 II.A.2.) not just one paragraph. If the specification provides even one embodiment of the claimed subject matter, then that may well suffice. Therefore, the question is not what paragraph [0038] states, but rather what the application as a whole states. This distinction is important because the examiner fails to provide any rationale for concluding that the application as a whole (e.g., the other portions of the written description) do provide sufficient descriptive support.

This misapplication of the possession inquiry is particularly glaring in the instant case because the examiner's comments about paragraph [0038][1] are logically inconsistent with the actual language of that paragraph and the rest of the specification. Paragraph [0038] clearly states that it is describeing some alternative embodiments, including alternatives to the example of FIG. 3. The examiner points to no rationale for concluding that the example of FIG. 3, itself, does not satisfy the possession requirement.

Third, it appears that the office action is confusing the specification for the claims. The claims set forth the bounds of the invention, not the specification. While the MPEP guides that examiners may refer to the specification, the examiners may not read the specification as setting forth the bounds of the protectable subject matter. To hold otherwise would violate a tenet of the patent law: "Limitations may not. . .be imported into the claims from the specification." MPEP § 2163, II.A.1.

If the examiner maintains this rejection, then the examiner is requested to memorialize for the record the specific bases for holding that the application as a whole falls short of 35 U.S.C. § 112, ¶ 1.

[1] [0038] Person of ordinary skill will appreciate that Figure 3 illustrates an example implementation only. Numerous alternatives may be made. For example, while a DEX Monitor is shown separately from a platform simulator, the two may be combined together. A DEX Monitor may monitor a Direct Execution Environment for any type of event, including non virtualization events. For example, with the example of FIG. 3, an instruction like a CPUID instruction may be executed in a Direct Execution Environment as a native instruction, or it can create a virtualization event, and be simulated in a software simulator (if it is desirable to have the simulated CPU be other than the host CPU). Further still, a DEX Monitor may switch between simulated virtual machines in a format other than a round robin format (e.g., giving one simulated CPU more execution quota than the others). Present Application.

3. REJECTION UNDER 35 U.S.C. § 112, ¶ 2

Claim 17 has been amended, thereby obviating the rejection of claim 17.

4. PRIOR ART CLAIM REJECTIONS

<u>Independent Claim 1</u>

Claim 1 has been amended above and now recites:

> 1. An article comprising a machine-accessible medium having stored thereon instructions that, when executed by a machine, cause the machine to:
>
> execute a host code in a host environment;
>
> create a plurality of virtual machines in a virtual environment, the virtual environment being a direct execution environment;
>
> transfer a virtual code from the host environment to the virtual environment; and
>
> execute virtual code on at least one of the plurality of virtual machines; and
>
> provide a single monitor within the host environment to control entry to and exit from each of the plurality of virtual machines in the direct execution environment.

The office action rejects claim 1 based on *Devine et al.*, which teaches a virtualization system that employs a virtual machine monitor (VMM) on segmented-architecture computers. The VMM monitors a virtual machine (VM) emulating the computer architecture, and determines whether code executing on the VM may be executed in a direct execution environment (e.g., where the host processor may be set up with reduced privileges) or must be executed in a binary translation environment (e.g., when the virtual and underlying architectures mismatch).

Devine et al., however, does not teach the recited subject and cannot because each VMM of *Devine et al.* is only capable of monitoring a **<u>single</u>** VM. Thus, *Devine et al.* does not teach creating "a plurality of virtual machines in a virtual environment, the virtual environment being a direct execution environment," executing "virtual code on at least one of the plurality of virtual machines" or providing "a monitor within the host environment to control entry to and exit from each of the plurality of virtual machines in the direct execution environment." As *Devine et* *al.* describes, each VMM is limited to supporting a single VM, such that if multiple VMs are desired then multiple VMMs must be employed:

> FIG. 7 shows one virtual machine monitor 100 supporting one virtual machine 120. The system according to the invention makes it possible to include any number of VMM's in a given implementation, **each supporting a corresponding VM**, limited only by available memory and speed requirements, and to switch between **the various included VMM's**. It is assumed merely for the sake of simplicity that the VMM 100 is the one actively in operation on the system. *Devine et al.* 24:18 – 26.

This requirement for a separate VMM for each VM is also described in the multiple processor emulation application, where *Devine et al.* explains that separate global and local shadow descriptors (one for each VMM) are required for each virtualized processor, thus preventing one monitor from handing multiple processor emulations:

> The use of multiple processors allows, for example, the simultaneous execution of multiple virtual machines, or the execution of a virtual machine with multiple virtual processors. **The virtualization of each processor is handled separately and independently**, including the decision as to whether to use direct execution or binary translation. For each virtual processor, the VMM will then maintain a separate set of global and local shadow descriptor entries. *Devine et al.* 25:7 – 14.

In short, *Devine et al.* does not (and cannot) teach or suggest the subject matter of claim 1.

The rejection of claim 1 is traversed and reconsideration requested. Claim 1 and claims 4 – 9 and newly-added claims 20 and 21 all depending therefrom are in condition for immediate allowance.

In addition to being in condition for allowance by dependency from claim 1, applicant separately highlights that *Devine et al.* also fails to teach the subject matter of claims 20 and 21. The prior art, for example, does not teach or suggest an apparatus capable of assigning "the virtual code to any one of the plurality of virtual machines for execution" or monitoring "the plurality of virtual machines for a virtualization event." Furthermore, the prior art does not teach or suggest determining "if the virtualization event is an end of quota event for one of the plurality of virtual machines" or "in response to an identification of the end of quota event, [switching] to another one of the plurality of virtual machines."

Independent Claim 11

Claim 11 has been amended above and now recites:

> 11. A method comprising:
>
> accessing simulated instruction codes in a host environment operating on a central processing unit (CPU) implementing Virtual Machine Extensions;
>
> launching a plurality of virtual machines in a virtual environment on the CPU, the virtual environment being a direct execution environment;
>
> virtualizing a CPU state associated with the simulated instruction codes;
>
> executing at least one of the simulated instruction codes on at least one of the plurality of virtual machines; and
>
> monitoring, via a single monitor, each of the plurality of virtual machines within the host environment to control entry to and exit from each of the plurality of virtual machines.

For at least the reasons outlined above with respect to claim 1, claim 11 is neither anticipated nor rendered obvious by the prior art. The rejection of claim 11 is traversed. Claim 11 and claims 12 – 16 and newly-added claim 22 depending therefrom are in condition for immediate allowance.

Independent Claim 17

Claim 17 has been amended above and now recites:

> 17. A system comprising:
>
> hardware to generate and control a plurality of virtual machines that each are capable of executing simulated instruction code, wherein the hardware is able to create an abstraction of a real machine for executing a real operating system on the computer system;
>
> a direct execution environment to execute the simulated instruction codes and associated data as virtual codes;

a plurality of virtual machines formed within the direct execution environment; and

a host environment comprising a single monitor for controlling exit from and entry to the plurality of virtual machines formed within the direction execution environment.

For at least the reasons outlined above – namely that *Devine et al.* does not teach or suggest a single monitor capable of controlling multiple virtual machines in a direct execution environment – claim 17 is neither anticipated nor rendered obvious by the prior art. The rejection of claim 17 is traversed. Claim 17 and claims 18 and 19 depending therefrom are in condition for immediate allowance.

In view of the above amendment, applicant believes the pending application is in condition for allowance.

Dated: November 10, 2006

Respectfully submitted,

By [illegible]
Paul B. Stephens
Registration No.: 47,970
MARSHALL, GERSTEIN & BORUN
233 S. Wacker Drive, Suite 6300
Sears Tower
Chicago, Illinois 60606-6357
(312) 474-6300
Attorney for Applicant